**Google mitigated the largest DDoS attack to date, peaking above 398 million rps**

October 10, 2023

#### **The attack used a novel technique, HTTP/2 Rapid Reset, based on stream multiplexing**

Over the last few years, Google's DDoS Response Team [has observed](https://cloud.google.com/blog/products/identity-security/identifying-and-protecting-against-the-largest-ddos-attacks) the trend that distributed denial-of-service (DDoS) attacks are increasing exponentially in size. Last year, we blocked the [largest DDoS attack](https://cloud.google.com/blog/products/identity-security/how-google-cloud-blocked-largest-layer-7-ddos-attack-at-46-million-rps) recorded at the time. This August, we stopped an even larger DDoS attack — 7½ times larger — that also used new techniques to try to disrupt websites and Internet services.

This new series of DDoS attacks reached a peak of 398 million requests per second (rps), and relied on a novel HTTP/2 “Rapid Reset” technique based on stream multiplexing that has affected multiple Internet infrastructure companies. By contrast, last year’s largest-recorded DDoS attack peaked at 46 million rps.

For a sense of scale, this two minute attack generated more requests than the total number of article views reported by Wikipedia during the entire month of September 2023.

Gráfico

Descripción generada automáticamente

*Google mitigated a DDoS attack which peaked at 398 million requests per second*

The most recent wave of attacks started in late August and continue to this day, targeting major infrastructure providers including Google services, Google Cloud infrastructure, and our customers. Although these attacks are among the largest attacks Google has seen, our global load-balancing and DDoS mitigation infrastructure helped keep our services running. In order to protect Google, our customers, and the rest of the Internet, we helped lead a coordinated effort with industry partners to understand the attack mechanics and collaborate on mitigations that can be deployed in response to these attacks.

La ola de ataques más reciente comenzó a finales de agosto y continúa hasta el día de hoy, dirigida a los principales proveedores de infraestructura, incluidos los servicios de Google, la infraestructura de Google Cloud y nuestros clientes. Aunque estos ataques se encuentran entre los ataques más grandes que Google haya visto, nuestra infraestructura global de equilibrio de carga y mitigación de DDoS ayudó a mantener nuestros servicios en funcionamiento. Para proteger a Google, a nuestros clientes y al resto de Internet, ayudamos a liderar un esfuerzo coordinado con socios de la industria para comprender la mecánica de los ataques y colaborar en las mitigaciones que se pueden implementar en respuesta a estos ataques.

Generally, DDoS attacks attempt to disrupt internet-facing websites and services, making them unreachable. Attackers direct overwhelming amounts of Internet traffic to targets, which can exhaust their ability to process incoming requests.

DDoS attacks can have wide-ranging impacts to victim organizations, including loss of business and unavailability of mission critical applications, which often cost victims time and money. Time to recover from DDoS attacks can stretch well beyond the end of an attack.

### Our investigation and response

Our investigation revealed that the attack was using a novel “Rapid Reset” technique that leverages stream multiplexing, a feature of the widely-adopted HTTP/2 protocol. We provide further analysis of this new Rapid Reset technique and discuss the evolution of Layer 7 attacks in a [companion blog](https://cloud.google.com/blog/products/identity-security/how-it-works-the-novel-http2-rapid-reset-ddos-attack).

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

*We observed the attack campaign continued over the course of September 2023*

We were able to mitigate the attack at the edge of Google's network, leveraging our significant investment in edge capacity to ensure our services and our customers’ services remained largely unaffected. As we understood more details about the attack methodology, we developed a set of mitigations and updated our proxies and denial-of-service defense systems to efficiently mitigate this technique. Since Google Cloud’s [Application Load Balancer](https://cloud.google.com/load-balancing?hl=en) and [Cloud Armor](https://cloud.google.com/armor/?hl=en) use the same hardware and software infrastructure that Google relies on to serve its own internet-facing services, the Cloud customers who use those services have their Internet-facing web apps and services similarly protected.

### Industry coordination and response for CVE-2023-44487

Soon after detecting the earliest of these attacks in August, Google applied additional mitigation strategies and coordinated a cross-industry response with other cloud providers and software maintainers who implement the HTTP/2 protocol stack. We shared intelligence about the attack and mitigation methodologies in real time as the attacks were underway.

This cross-industry collaboration has resulted in patches and other mitigation techniques used by many large infrastructure providers. The collaboration helped to pave the way for today’s coordinated responsible disclosure of the new attack methodology and potential susceptibility across a multitude of common open source and commercial proxies, application servers, and load balancers.

Poco después de detectar el primero de estos ataques en agosto, Google aplicó estrategias de mitigación adicionales y coordinó una respuesta intersectorial con otros proveedores de nube y mantenedores de software que implementan la pila de protocolos HTTP/2. Compartimos inteligencia sobre el ataque y las metodologías de mitigación en tiempo real mientras los ataques estaban en marcha.

Esta colaboración entre industrias ha dado como resultado parches y otras técnicas de mitigación utilizadas por muchos grandes proveedores de infraestructura. La colaboración ayudó a allanar el camino para la divulgación responsable coordinada de hoy de la nueva metodología de ataque y la susceptibilidad potencial a través de una multitud de servidores de aplicaciones, servidores de aplicaciones y balanceadores de carga comunes de código abierto y comerciales.

The collective susceptibility to this attack is being tracked as [CVE-2023-44487](https://nvd.nist.gov/vuln/detail/CVE-2023-44487) and has been designated a High severity vulnerability with a [CVSS](https://nvd.nist.gov/vuln-metrics/cvss) score of 7.5 (out of 10).

Google expresses sincere gratitude to all of the cross-industry stakeholders who have collaborated, shared information, accelerated patching of their infrastructure, and rapidly made patches available to their customers.

Google expresa su más sincero agradecimiento a todas las partes interesadas de todas las industrias que han colaborado, compartido información, acelerado la aplicación de parches a su infraestructura y rápidamente han puesto parches a disposición de sus clientes.

### Who is susceptible and what to do about it

Any enterprise or individual that is serving an HTTP-based workload to the Internet may be at risk from this attack. Web applications, services, and APIs on a server or proxy able to communicate using the HTTP/2 protocol could be vulnerable. Organizations should verify that any servers they run that support HTTP/2 are not vulnerable, or apply vendor patches for CVE-2023-44487 to limit impact from this attack vector. If you are managing or operating your own HTTP/2-capable server (open source or commercial) you should immediately apply a patch from the relevant vendor when available.

### Next steps

Defending against massive DDoS attacks such as those described here is difficult. With or without patches, organizations would need to make significant infrastructure investments to keep services running in the face of attacks of any moderate size and larger. Instead of bearing that expense themselves, organizations running services on Google Cloud can take advantage of our investment in capacity at global scale in our [Cross-Cloud Network](https://cloud.google.com/solutions/cross-cloud-network) to deliver and protect their applications.

Google Cloud customers exposing their services using the global or regional Application Load Balancer benefit from [Cloud Armor always-on DDoS protection](https://cloud.google.com/armor), where attacks exploiting vulnerabilities such as CVE-2023-44487 are quickly mitigated.

Even though with Cloud Armor always-on DDoS protection we are able to efficiently absorb most of the hundreds of millions of requests per second at the edge of Google’s network, millions of unwelcome requests per second can still make it through. To protect against this and other layer 7 attacks, we also recommend deployment of Cloud Armor [custom security policies](https://cloud.google.com/armor/docs/best-practices) with proactive [rate limiting](https://cloud.google.com/armor/docs/rate-limiting-overview) rules and AI-powered [Adaptive Protection](https://cloud.google.com/armor/docs/adaptive-protection-overview) to more comprehensively detect, analyze, and mitigate attack traffic.

We provide more technical information on this [current wave of DDoS attacks here](https://cloud.google.com/blog/products/identity-security/how-it-works-the-novel-http2-rapid-reset-ddos-attack), and you can learn more about Google Cloud Armor’s DDoS protection [here](https://cloud.google.com/armor).

**How it works: The novel HTTP/2 ‘Rapid Reset’ DDoS attack**

A number of Google services and Cloud customers have been targeted with a novel HTTP/2-based DDoS attack which peaked in August. These attacks were significantly larger than any [previously-reported Layer 7 attacks](https://cloud.google.com/blog/products/identity-security/how-google-cloud-blocked-largest-layer-7-ddos-attack-at-46-million-rps), with the largest attack [surpassing 398 million requests per second](https://cloud.google.com/blog/products/identity-security/google-cloud-mitigated-largest-ddos-attack-peaking-above-398-million-rps).

The attacks were largely stopped at the edge of our network by Google's global load balancing infrastructure and did not lead to any outages. While the impact was minimal, Google's DDoS Response Team reviewed the attacks and added additional protections to further mitigate similar attacks. In addition to Google's internal response, we helped lead a coordinated disclosure process with industry partners to address the new HTTP/2 vector across the ecosystem.

Los ataques fueron detenidos en gran medida en el borde de nuestra red gracias a la infraestructura de equilibrio de carga global de Google y no provocaron ninguna interrupción. Si bien el impacto fue mínimo, el equipo de respuesta DDoS de Google revisó los ataques y agregó protecciones adicionales para mitigar aún más ataques similares. Además de la respuesta interna de Google, ayudamos a liderar un proceso de divulgación coordinado con socios de la industria para abordar el nuevo vector HTTP/2 en todo el ecosistema.

Below, we explain the predominant methodology for Layer 7 attacks over the last few years, what changed in these new attacks to make them so much larger, and the mitigation strategies we believe are effective against this attack type. This article is written from the perspective of a reverse proxy architecture, where the HTTP request is terminated by a reverse proxy that forwards requests to other services. The same concepts apply to HTTP servers that are integrated into the application server, but with slightly different considerations which potentially lead to different mitigation strategies.

### A primer on HTTP/2 for DDoS

Since late 2021, the majority of Layer 7 DDoS attacks we've observed across Google first-party services and Google Cloud projects protected by [Cloud Armor](https://cloud.google.com/armor) have been based on HTTP/2, both by number of attacks and by peak request rates.

A primary design goal of HTTP/2 was efficiency, and unfortunately the features that make HTTP/2 more efficient for legitimate clients can also be used to make DDoS attacks more efficient.

### Stream multiplexing

HTTP/2 uses "streams", bidirectional abstractions used to transmit various messages, or "frames", between the endpoints. “Stream multiplexing” is the core HTTP/2 feature which allows higher utilization of each TCP connection. Streams are multiplexed in a way that can be tracked by both sides of the connection while only using one Layer 4 connection. Stream multiplexing enables clients to have multiple in-flight requests without managing multiple individual connections.

One of the main constraints when mounting a Layer 7 DoS attack is the number of concurrent transport connections. Each connection carries a cost, including operating system memory for socket records and buffers, CPU time for the TLS handshake, as well as each connection needing a unique four-tuple, the IP address and port pair for each side of the connection, constraining the number of concurrent connections between two IP addresses.

In HTTP/1.1, each request is processed serially. The server will read a request, process it, write a response, and only then read and process the next request. In practice, this means that the rate of requests that can be sent over a single connection is one request per round trip, where a round trip includes the network latency, proxy processing time and backend request processing time. While HTTP/1.1 pipelining is available in some clients and servers to increase a connection's throughput, it is not prevalent amongst legitimate clients.

With HTTP/2, the client can open multiple concurrent streams on a single TCP connection, each stream corresponding to one HTTP request. The maximum number of concurrent open streams is, in theory, controllable by the server, but in practice clients may open 100 streams per request and the servers process these requests in parallel. It’s important to note that server limits can not be unilaterally adjusted.

For example, the client can open 100 streams and send a request on each of them in a single round trip; the proxy will read and process each stream serially, but the requests to the backend servers can again be parallelized. The client can then open new streams as it receives responses to the previous ones. This gives an effective throughput for a single connection of 100 requests per round trip, with similar round trip timing constants to HTTP/1.1 requests. This will typically lead to almost 100 times higher utilization of each connection.

### The HTTP/2 Rapid Reset attack

The HTTP/2 protocol allows clients to indicate to the server that a previous stream should be canceled by sending a RST\_STREAM frame. The protocol does not require the client and server to coordinate the cancellation in any way, the client may do it unilaterally. The client may also assume that the cancellation will take effect immediately when the server receives the RST\_STREAM frame, before any other data from that TCP connection is processed.

This attack is called Rapid Reset because it relies on the ability for an endpoint to send a RST\_STREAM frame immediately after sending a request frame, which makes the other endpoint start working and then rapidly resets the request. The request is canceled, but leaves the HTTP/2 connection open.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

*HTTP/1.1 and HTTP/2 request and response pattern*

The HTTP/2 Rapid Reset attack built on this capability is simple: The client opens a large number of streams at once as in the standard HTTP/2 attack, but rather than waiting for a response to each request stream from the server or proxy, the client cancels each request immediately.

The ability to reset streams immediately allows each connection to have an indefinite number of requests in flight. By explicitly canceling the requests, the attacker never exceeds the limit on the number of concurrent open streams. The number of in-flight requests is no longer dependent on the round-trip time (RTT), but only on the available network bandwidth.

In a typical HTTP/2 server implementation, the server will still have to do significant amounts of work for canceled requests, such as allocating new stream data structures, parsing the query and doing header decompression, and mapping the URL to a resource. For reverse proxy implementations, the request may be proxied to the backend server before the RST\_STREAM frame is processed. The client on the other hand paid almost no costs for sending the requests. This creates an exploitable cost asymmetry between the server and the client.

Another advantage the attacker gains is that the explicit cancellation of requests immediately after creation means that a reverse proxy server won't send a response to any of the requests. Canceling the requests before a response is written reduces downlink (server/proxy to attacker) bandwidth.

### HTTP/2 Rapid Reset attack variants

In the weeks after the initial DDoS attacks, we have seen some Rapid Reset attack variants. These variants are generally not as efficient as the initial version was, but might still be more efficient than standard HTTP/2 DDoS attacks.

The first variant does not immediately cancel the streams, but instead opens a batch of streams at once, waits for some time, and then cancels those streams and then immediately opens another large batch of new streams. This attack may bypass mitigations that are based on just the rate of inbound RST\_STREAM frames (such as allow at most 100 RST\_STREAMs per second on a connection before closing it).

These attacks lose the main advantage of the canceling attacks by not maximizing connection utilization, but still have some implementation efficiencies over standard HTTP/2 DDoS attacks. But this variant does mean that any mitigation based on rate-limiting stream cancellations should set fairly strict limits to be effective.

The second variant does away with canceling streams entirely, and instead optimistically tries to open more concurrent streams than the server advertised. The benefit of this approach over the standard HTTP/2 DDoS attack is that the client can keep the request pipeline full at all times, and eliminate client-proxy RTT as a bottleneck. It can also eliminate the proxy-server RTT as a bottleneck if the request is to a resource that the HTTP/2 server responds to immediately.

[RFC 9113](https://www.rfc-editor.org/rfc/rfc9113.html), the current HTTP/2 RFC, suggests that an attempt to open too many streams should invalidate only the streams that exceeded the limit, not the entire connection. We believe that most HTTP/2 servers will not process those streams, and is what enables the non-cancelling attack variant by almost immediately accepting and processing a new stream after responding to a previous stream.

### A multifaceted approach to mitigations

We don't expect that simply blocking individual requests is a viable mitigation against this class of attacks — instead the entire TCP connection needs to be closed when abuse is detected. HTTP/2 provides built-in support for closing connections, using the GOAWAY frame type. The RFC defines a process for gracefully closing a connection that involves first sending an informational GOAWAY that does not set a limit on opening new streams, and one round trip later sending another that forbids opening additional streams.

However, this graceful GOAWAY process is usually not implemented in a way which is robust against malicious clients. This form of mitigation leaves the connection vulnerable to Rapid Reset attacks for too long, and should not be used for building mitigations as it does not stop the inbound requests. Instead, the GOAWAY should be set up to limit stream creation immediately.

This leaves the question of deciding which connections are abusive. The client canceling requests is not inherently abusive, the feature exists in the HTTP/2 protocol to help better manage request processing. Typical situations are when a browser no longer needs a resource it had requested due to the user navigating away from the page, or applications using a [long polling](https://en.wikipedia.org/wiki/Push_technology" \l "Long_polling" \t "_blank) approach with a client-side timeout.

Mitigations for this attack vector can take multiple forms, but mostly center around tracking connection statistics and using various signals and business logic to determine how useful each connection is. For example, if a connection has more than 100 requests with more than 50% of the given requests canceled, it could be a candidate for a mitigation response. The magnitude and type of response depends on the risk to each platform, but responses can range from forceful GOAWAY frames as discussed before to closing the TCP connection immediately.

To mitigate against the non-cancelling variant of this attack, we recommend that HTTP/2 servers should close connections that exceed the concurrent stream limit. This can be either immediately or after some small number of repeat offenses.

### Applicability to other protocols

We do not believe these attack methods translate directly to HTTP/3 (QUIC) due to protocol differences, and Google does not currently see HTTP/3 used as a DDoS attack vector at scale. Despite that, our recommendation is for HTTP/3 server implementations to proactively implement mechanisms to limit the amount of work done by a single transport connection, similar to the HTTP/2 mitigations discussed above.

### Industry coordination

Early in our DDoS Response Team's investigation and in coordination with industry partners, it was apparent that this new attack type could have a broad impact on any entity offering the HTTP/2 protocol for their services. Google helped lead a coordinated vulnerability disclosure process taking advantage of a pre-existing coordinated vulnerability disclosure group, which has been used for a number of other efforts in the past.

During the disclosure process, the team focused on notifying large-scale implementers of HTTP/2 including infrastructure companies and server software providers. The goal of these prior notifications was to develop and prepare mitigations for a coordinated release. In the past, this approach has enabled widespread protections to be enabled for service providers or available via software updates for many packages and solutions.

During the coordinated disclosure process, we reserved [CVE-2023-44487](https://nvd.nist.gov/vuln/detail/CVE-2023-44487) to track fixes to the various HTTP/2 implementations.

### Next steps

The novel attacks discussed in this post can have significant impact on services of any scale. All providers who have HTTP/2 services should assess their exposure to this issue. Software patches and updates for common web servers and programming languages may be available to apply now or in the near future. We recommend applying those fixes as soon as possible.

For our customers, we recommend patching software and enabling the [Application Load Balancer](https://cloud.google.com/load-balancing?hl=en) and [Google Cloud Armor](https://cloud.google.com/armor), which has been protecting Google and existing Google Cloud Application Load Balancing users.

REFLEXIONES:

La seguridad en redes es crucial en la era digital actual, donde la información y los servicios vitales están interconectados a través de redes. Algunas reflexiones clave sobre la importancia de la seguridad en redes incluyen:

Interconexión Global:

La mayoría de las organizaciones y servicios dependen de redes para operar y compartir información en una escala global. Esto hace que la seguridad en redes sea esencial para proteger la confidencialidad, integridad y disponibilidad de los datos.

Aumento de las Amenazas:

Con el aumento de la conectividad, las amenazas a la seguridad también han evolucionado. Desde ataques DDoS hasta intrusiones más sofisticadas, la diversidad de amenazas exige un enfoque integral para proteger las redes.

Protección de la Privacidad:

La información personal y empresarial se transmite a través de redes, y la seguridad es fundamental para proteger la privacidad de los individuos y la propiedad intelectual de las organizaciones.

Consecuencias Financieras y Operativas:

Las violaciones de seguridad en redes pueden tener consecuencias financieras significativas, que van desde pérdidas económicas directas hasta la disminución de la confianza de los clientes. Además, pueden interrumpir operaciones críticas.

Colaboración y Confianza:

La colaboración entre empresas, gobiernos y usuarios finales depende en gran medida de la confianza en la seguridad de las redes. La pérdida de confianza puede afectar negativamente la adopción de nuevas tecnologías y servicios.

Para mitigar las vulnerabilidades en las redes, es crucial contar con una estrategia integral:

Educación y Concienciación:

La concienciación y educación de los usuarios y profesionales de TI son fundamentales para prevenir ataques mediante prácticas seguras y la comprensión de las amenazas actuales.

Implementación de Mejores Prácticas:

Las organizaciones deben seguir las mejores prácticas de seguridad, como el cifrado de datos, la autenticación fuerte, y la aplicación de actualizaciones y parches de seguridad.

Infraestructuras de Seguridad:

Utilizar firewalls, sistemas de detección y prevención de intrusiones, y herramientas de análisis de tráfico son elementos clave para construir una infraestructura segura.

Colaboración y Coordinación:

La colaboración entre las partes interesadas, incluidos proveedores de servicios, fabricantes de hardware y software, y organismos gubernamentales, es esencial para abordar las amenazas de manera efectiva.

Monitoreo y Respuesta Rápida:

La capacidad de monitorear las redes en tiempo real y responder rápidamente a las amenazas es esencial. Los equipos de seguridad deben estar preparados para identificar y contener posibles violaciones.

Diseñar sistemas con la seguridad en mente desde el principio es esencial para reducir la superficie de ataque y anticipar posibles amenazas. Esto implica realizar evaluaciones de riesgos, implementar medidas de seguridad desde el diseño inicial, y continuar mejorando las defensas a medida que evolucionan las amenazas. La seguridad debe ser un componente integral en todos los aspectos del desarrollo y mantenimiento de sistemas y redes.

HTTP/2 Rapid Reset:

El HTTP/2 Rapid Reset se refiere a una técnica de ataque DDoS (Denegación de Servicio Distribuido) que se basa en la capacidad del protocolo HTTP/2 para cancelar rápidamente transmisiones después de enviar una solicitud. En este tipo de ataque, el atacante abre múltiples flujos de solicitud a la vez y, en lugar de esperar respuestas del servidor, cancela cada solicitud de inmediato. Esto permite al atacante mantener un gran número de transmisiones abiertas simultáneamente, evitando límites tradicionales y agotando recursos del servidor.

Multiplexación de Flujo:

La multiplexación de flujo es una característica clave del protocolo HTTP/2 que permite la transmisión simultánea de múltiples mensajes (o flujos) entre un cliente y un servidor a través de una única conexión TCP. Esto mejora la eficiencia y el rendimiento de las comunicaciones web, ya que permite que múltiples solicitudes y respuestas se transmitan en paralelo. Cada flujo está identificado por un identificador único y puede ser rastreado por ambos extremos de la conexión.

Capas en los Artículos (Capa 7 y Capa 4):

En el contexto de las redes, se hace referencia a diferentes capas del modelo OSI (Open Systems Interconnection). En los artículos, se menciona la "Capa 7" y la "Capa 4":

Capa 7 (Capa de Aplicación): Se refiere a la capa más alta del modelo OSI, donde ocurren las interacciones de usuario y las aplicaciones. En el contexto de los ataques DDoS, los ataques a la "Capa 7" se centran en servicios de aplicación, como ataques a sitios web o servicios web.

Capa 4 (Capa de Transporte): Se refiere a la capa que maneja el transporte de datos entre sistemas. En este contexto, se habla de la Capa 4 para referirse a la conexión TCP (Transmission Control Protocol) utilizada para la comunicación entre el cliente y el servidor.

Estos conceptos están relacionados con la arquitectura de las redes y la forma en que los protocolos como HTTP/2 gestionan las comunicaciones entre clientes y servidores. El HTTP/2 Rapid Reset aprovecha las características específicas de HTTP/2, como la multiplexación de flujo, para llevar a cabo ataques DDoS más eficaces en la Capa 7 de la red. La comprensión de estas capas y protocolos es esencial para abordar y defenderse contra amenazas de seguridad en redes.